An Administrative Tool for Distributed Security Task Scheduling

Greg Dunlap and Dipankar Dasgupta
Division of Computer Science
The Department of Mathematical Sciences
The University of Memphis
Memphis, TN 38152
gdunlap@acm.org, ddasgupt@memphis.edu

Abstract:
We developed an administrative tool for Distributed Security Scheduling of various security tasks by providing a framework for integration, coordination, and resource sharing of several security technologies. It serves as a wrapper for commercial-off-the-shelf (COTS) security software. This tool will be useful for security administrators for specific security measures and to execute appropriate security tools remotely in a coordinated fashion.

Introduction

There are many software packages that are available for computer systems to detect and report potential or existing computer system security irregularities. These irregularities could be linked to attempted or successful attacks, which may result in system failure or compromise. Since there is no single or simple general security technology, and security breaches could be internal, external, accidental, or intentional, there are a variety of tools available to help protect a system (file integrity checkers, virus scanners, intrusion detectors, port scanners, etc.). Each package has a specific purpose, some may overlap, while most don't. Each package has strengths and weaknesses that can potentially be exploited [Ko et al. 1997].

Most of these packages operate independently, without data exchange or consistent security policies. Each package may have been written by a different vendor, perhaps even competitors in the industry. Since there is no consistent data exchange between these packages, administrator intervention is usually required to analyze the acquired data and make decisions about what actions may need to be taken to prevent a compromise, or to recover from one. Administrator action can however be too late, or not at all. This may be because the compromise was too quick for the administrator to respond, or the administrator failed to recognize what was happening. Therefore, existing security measures not only depend on the available tools, but also on properly trained administration personnel to execute the appropriate security countermeasures. As a result, some computer systems are inherently less secure than others. There have been many approaches to a more secure system and this work is based on an understanding many of these. [Premkumar 1998, Moriconi 1997, Wulf 1995]

The premise behind the Distributed Security Scheduler (DSS) is to allow the average user to schedule multiple security applications on their computer, and determine a predefined run time or manually run the application at a time of their choosing. This allows the user to rest assured that the programs will run at the appropriate time. Each DSS has a server built into it so that the user or administrator can remotely login to the DSS and change the time or method by which a program is launched. This administrative tool is called the Distributed Security Scheduler Administrator (DSSA).

Proposed Method

The DSS and DSSA are written in C++ and their graphical user interface (GUI) is built with the QT 2.3.1 libraries [Trolltech 2001]. QT is a cross platform C++ library set that has classes for GUI's and other data structures (string, linked list, queue, etc) that allows the developers to write code with a GUI without having to worry with the platform concerns, the same code will compile on Linux, Solaris, Mac OS X, or Windows with only minor code changes where file structure is concerned. This

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE (DD-MM-YYYY) 25-01-2002	2. REPORT TYPE	3. DATES COVERED (FROM - TO) xx-xx-2002 to xx-xx-2002

4. TITLE AND SUBTITLE An Administrative Tool for Distributed Security Task Scheduling Unclassified	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Dunlap, Greg ; Dasgupta, Dipankar ;	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME AND ADDRESS The University of Memphis Memphis, TN38152	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME AND ADDRESS IATAC 3190 Fairview Park Drive Falls Church, VA22042	10. SPONSOR/MONITOR'S ACRONYM(S) 11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APUBLIC RELEASE
,

13. SUPPLEMENTARY NOTES

14. ABSTRACT
See report.

15. SUBJECT TERMS
IATAC COLLECTION

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Public Release	18. NUMBER OF PAGES 9	19. NAME OF RESPONSIBLE PERSON email from Booz Allen Hamilton (IATAC), (blank) lfenster@dtic.mil
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			19b. TELEPHONE NUMBER International Area Code Area Code Telephone Number 703767-9007 DSN 427-9007

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39.18

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** 1/25/2002	**3. REPORT TYPE AND DATES COVERED** Report 1/25/2002

4. TITLE AND SUBTITLE An Administrative Tool for Distributed Security Task Scheduling	**5. FUNDING NUMBERS**
6. AUTHOR(S) Dunlap, Greg; Dasgupta, Dipankar	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) The University of Memphis Memphis, TN 38152	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) IATAC 3190 Fairview Park Drive Falls Church, VA 22042	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; Distribution unlimited	**12b. DISTRIBUTION CODE** A

13. ABSTRACT *(Maximum 200 Words)*

We developed an administrative tool for Distributed Security Scheduling of various security tasks by providing a framework for integration, coordination, and resource sharing of several security technologies. It serves as a wrapper for commercial-off-the-shelf (COTS) security software. This tool will be useful for security administrators for specific security measures and to execute appropriate security tools remotely in a coordinated fashion.

14. SUBJECT TERMS IATAC Collection, information security, security tasks, COTS	**15. NUMBER OF PAGES** 8
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	**18. SECURITY CLASSIFICATION OF THIS PAGE** UNCLASSIFIED	**19. SECURITY CLASSIFICATION OF ABSTRACT** UNCLASSIFIED	**20. LIMITATION OF ABSTRACT** UNLIMITED

way a Windows computer network can have the schedulers installed on them, and the administrator tool can be running on a Linux box and still be able to commutate with them with out any problems.

By using the QT tool set the size of the GUI application code is reduced. We can gain a lot of information about the complexity of the code by looking at the Halstead metrics [Pressman 2001] for the source code. Table 1 shows the Halstead variables used for calculation, and table 2 shows the metrics and how they are computed from the variables.

Table 1 Halstead variables

n1	=	the number of distinct operators
n2	=	the number of distinct operands
N1	=	the total number of operators
N2	=	the total number of operands

Table 2 Halstead metrics

Measure	Symbol	Formula
Program length	N	N= N1 + N2
Program vocabulary	n	n= n1 + n2
Volume	V	V= N * (LOG2 n)

From this we can derive the Halstead metrics for the source code written. Table 3 shows the metrics per file, plus the logical lines of code for the original code written for the DSS.

Table 3: New Code written for DSS

Filename	LOC	n	N	V
schedular.h	74	82	202	1284
schedular.cpp	267	184	1364	10262
execute.h	54	54	122	702
execute.cpp	170	78	968	6084
configdialog.h	230	167	611	4511
configdialog.cpp	1328	367	8163	69545
server.h	24	40	62	329
server.cpp	23	44	102	556
clientsocket.h	25	39	65	343
clientsocket.cpp	743	164	4285	31527
main.cpp	16	28	56	269
Totals	2954	1247	16000	125412

Table 4 gives the metrics for the code written for the DSSA.

Table 4: New Code written for DSSA

Filename	LOC	n	N	V
clientgui.h	56	72	155	956
cleintgui.cpp	245	158	1480	10809
configclientdialog.h	118	119	338	2330
configclientdialog.cpp	1252	334	7451	62466
hostdialog.h	29	50	79	445
hostdialog.cpp	56	74	264	1639
main.cpp	16	32	60	300
Totals	1772	839	9827	78945

The DSS consist of 8 classes. Three of which are the string classes and associated classes. The main class is the scheduler class, this is the main GUI for the program, it has a server class and an execute class. The execute class is the code that handles the spawning of the scheduled programs. This is the only class that had to be modified for each operation system because of the differences in file systems and launching code. The server that the scheduler is a background server, which passively listens for new TCP connections, upon receiving a new connection, the server spawns of a client socket class, which handles the interactions with the client. All three of these classes have a dependency link to an instance of the configuration dialog, this way all of them can share the same information. A UML diagram [Booch 1999] of this can be seen in figure 1.

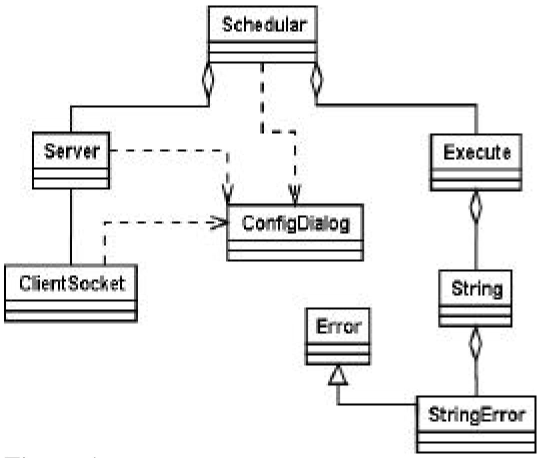


Figure 1

The DSSA is a simple program that has three classes. The client GUI handles the main interface (the three options the user has to run the DSSA). It has a host dialog, and a configuration dialog. If the user chooses the first radio button then the client GUI immediately launches the configuration dialog for the client. If the user selects on of the other radio buttons, then the client GUI turns over control to the host dialog which builds the popup box with the hosts or IP address, when the user double clicks on of them it will launch a configuration dialog for that host or IP.

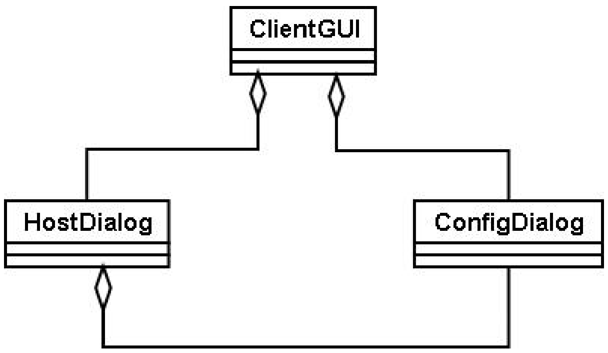
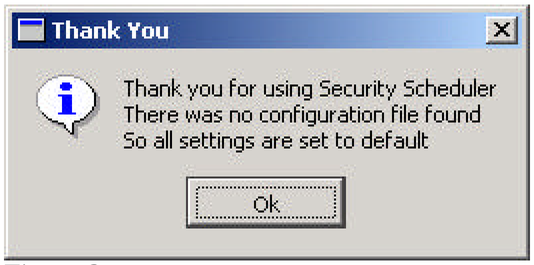


Figure 2

Experimentation

When the user first starts up the scheduler and there is no configuration file present, they will receive a dialog box (see figure 3) telling them that no configuration file was found and that all settings are being set to default. This has two purposes, the first is that this allows us to thank the user for using the product, and alerts existing users that the configuration file was not found.


Thank You

Thank you for using Security Scheduler
There was no configuration file found
So all settings are set to default

Figure 3

The next thing the user sees is the main scheduler window in its default mode (see figure 4). There are places for five programs to be scheduled and each of them has a button associated with it, the button name is by default Progx, where x is the button number.

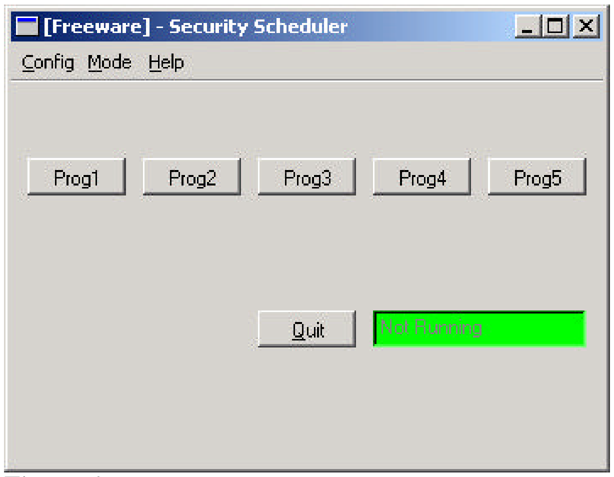
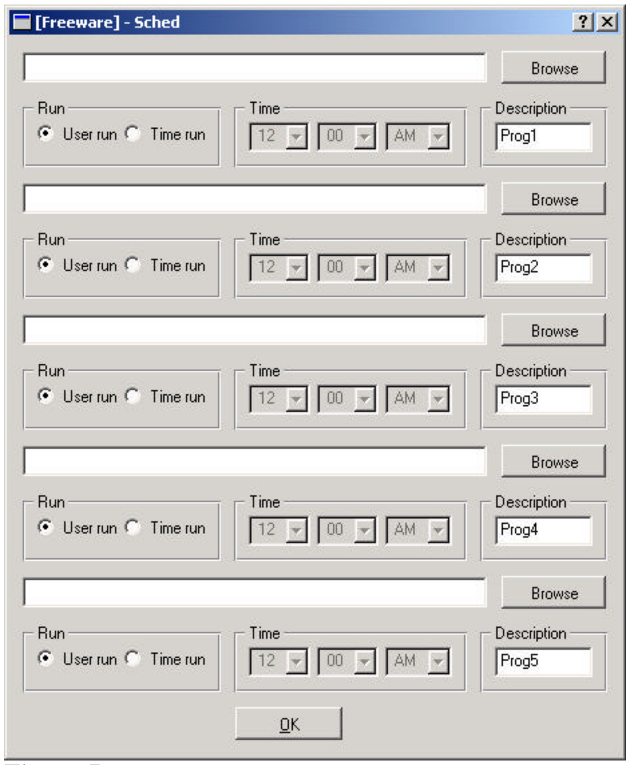


Figure 4

There are three menu items on the menu bar; the first is the configuration menu, which allows the user to configure the applications to run, their description, their run mode, and time. The mode option allows the user to specify what state the program is in. There are two modes, running and stopped. In run mode the program will scheduler all time dependent applications and launch them at the correct time. In stopped mode all timers are stopped. There is a color-associated box at the bottom right to tell the user what mode the application is currently in. Green is for not running and red if for running, also the words will appear in the box as well. The third menu item is the help box that displays information about the program. The quit button on the bottom of the application is the preferred why to exit.

The configure dialog box (see figure 5) is launched when the configure pull down menu is selected from the main application and activated.



Run	Time	Description
[✓ User run] Time run	12 : 00 AM	Prog1
[✓ User run] Time run	12 : 00 AM	Prog2
[✓ User run] Time run	12 : 00 AM	Prog3
[✓ User run] Time run	12 : 00 AM	Prog4
[✓ User run] Time run	12 : 00 AM	Prog5

Figure 5

Here the user can configure the scheduler. For each application the process is the same. The long edit box is the absolute path for the program to schedule; the browse button will launch a file dialog (see figure 6) that will let them search for the program they want to find. This is analogous to the common file open dialog seen in most Windows ™ applications.

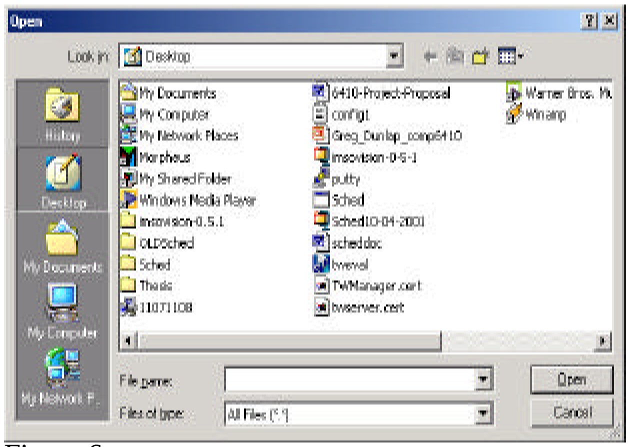
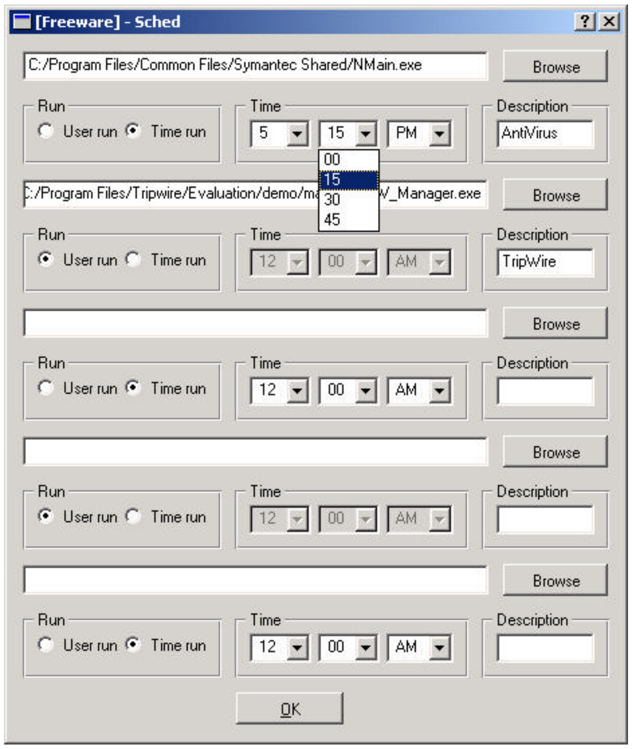


Figure 6

Once the user has selected the application they want to run, they must decide on the run state of it. The choices are user run, or time run. The default is user run, this means that the user decides when to run the application by clicking the program button associated with it in the main application window. If the user selects time run, the combo boxes that specify time become activated and the user can select and hour, 00, 15, 30, or 45 in the minute box, and AM or PM (see figure 7). Once that is complete the user can enter a description of the program in the description field. By default it is Progx as seen in the main application. The description the user enters in will become the name that is on the button associated with it in the main application (see figure 8). Clicking the OK button will save all the configuration changes and update the main application.



Path	Run	Time	Description
C:/Program Files/Common Files/Symantec Shared/NMain.exe	User run [✓ Time run]	5 : 15 PM	AntiVirus
:/Program Files/Tripwire/Evaluation/demo/ma W_Manager.exe	[✓ User run] Time run	12 : 00 AM	TripWire
	User run [✓ Time run]	12 : 00 AM	
	[✓ User run] Time run	12 : 00 AM	
	User run [✓ Time run]	12 : 00 AM	

Figure 7

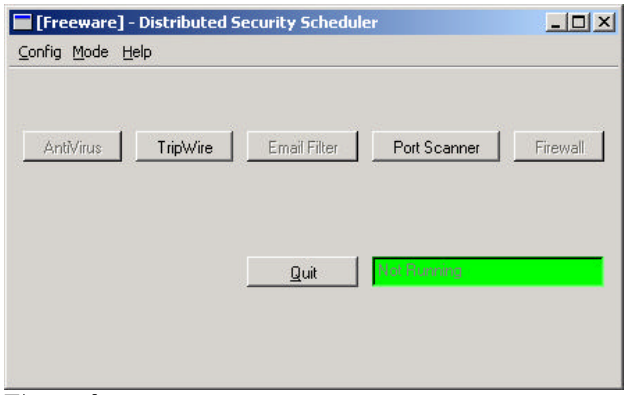


Figure 8

Notice that the AntiVirus button has been deactivate, this is because it is time dependent, the IE button is still user click able so it is enabled (see figure 8).

To put the program into run mode the user will click on the Mode pull down menu and select run, this will turn the status box to red and it will say Running. (See figure 9)

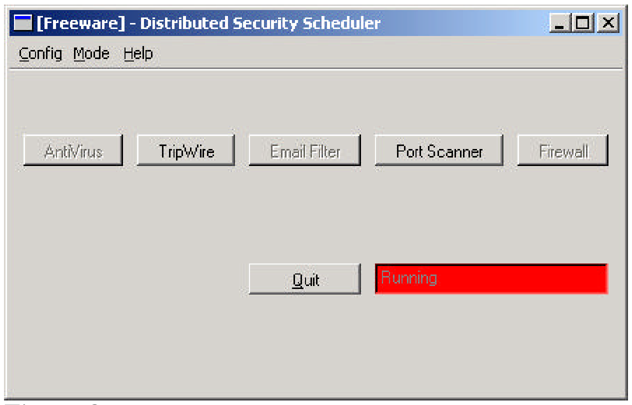
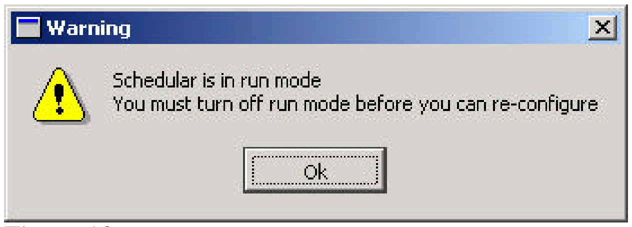


Figure 9

This causes the program to calculate how much time there is between the system clock and the time the application needs to run, this is then converted to m-sec and the timer is set.
If the user tries to re-configure the application while it is in run mode they will get a warning box (see figure 10) telling them to first switch to the stop mode. If there are no programs that are time run, then the user cannot switch the program to run mode.


Warning

Schedular is in run mode
You must turn off run mode before you can re-configure

Figure 10

When the program is started and there is a configuration file found and there are applications that are time dependent, then the application will immediately go into run mode.

When the user starts up the DSSA they will see a GUI like figure 11. This gives the administrator three different choices as seen by the radio buttons.

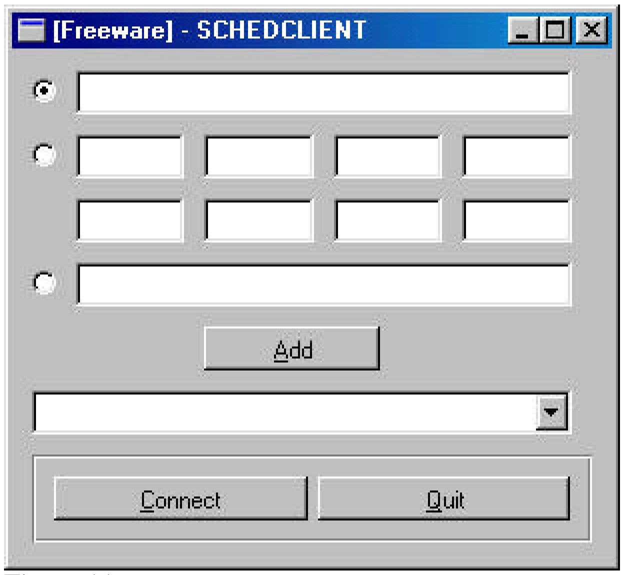
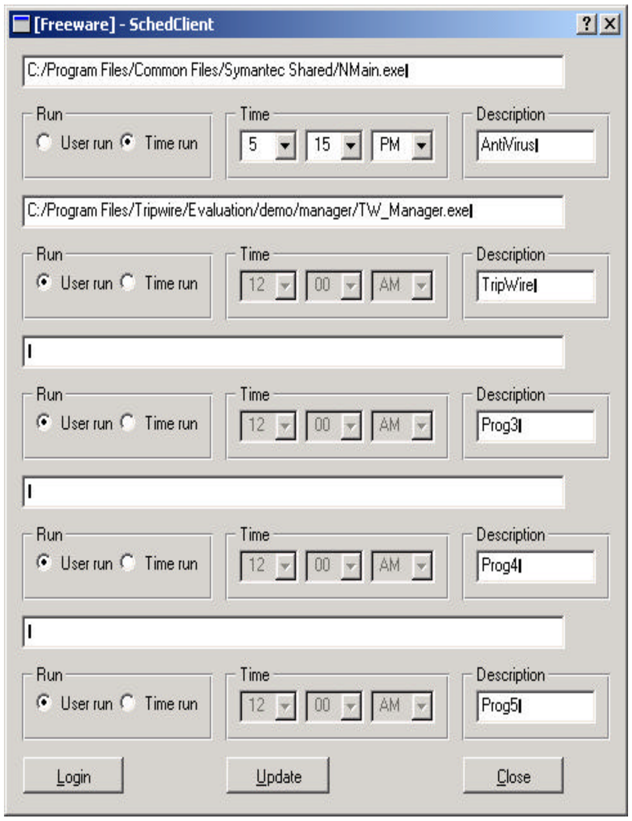


Figure 11

If the first radio button is selected then the administrator is selecting to just look at a single host. They would then type in the name of the host, or the IP address and hit the connect button. If the connection is successful then they will open up a configuration dialog similar to figure 5, but with a few differences. Instead of an OK button, there are Login, Update, and Close buttons. By clicking Login, the user is requesting the configuration information from the server it is connected to (see figure 12).



C:/Program Files/Common Files/Symantec Shared/NMain.exe

Run: User run [✓ Time run] — Time: 5 15 PM — Description: AntiVirus

C:/Program Files/Tripwire/Evaluation/demo/manager/TW_Manager.exe

Run: [✓ User run] Time run — Time: 12 00 AM — Description: TripWire

Run: [✓ User run] Time run — Time: 12 00 AM — Description: Prog3

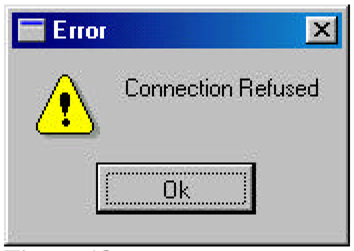
Run: [✓ User run] Time run — Time: 12 00 AM — Description: Prog4

Run: [✓ User run] Time run — Time: 12 00 AM — Description: Prog5

Figure 12

As the user makes changes to the configuration (user or time run, time to run, description) they can update the DSS they are connected to by clicking the Update button. This will immediately apply any changed to the DSS. When the user is done they can click the Close button to close the configuration dialog and close the socket connection.

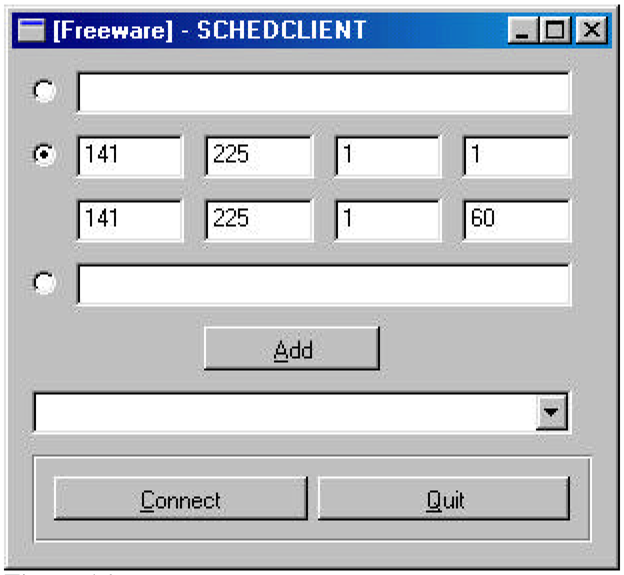
If there is a connection problem when the connect button is clicked a pop-up dialog will appear like figure 13 letting the user know what the problem is.



Connection Refused

Figure 13

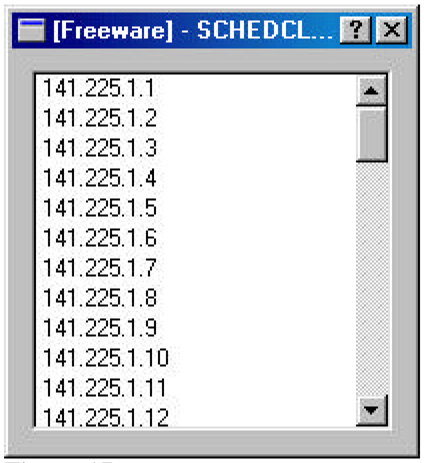
The second radio button on the DSSA allows the user to specify a range of IP addresses to which they want to connect to. So by using the boxes provided they user can enter in a sequence like figure 14.



141	225	1	1
141	225	1	60

Figure 14

This will provide a pop-up like figure 15.



141.225.1.1
141.225.1.2
141.225.1.3
141.225.1.4
141.225.1.5
141.225.1.6
141.225.1.7
141.225.1.8
141.225.1.9
141.225.1.10
141.225.1.11
141.225.1.12

Figure 15

By double clicking on one of the IP addresses the user can then open up a configuration dialog for that IP address (see figure 12).

The third radio button on the DSSA allows the user to input a list of host names or IP address they want to specifically look at. In the line edit the user inputs the host name or IP address and then click Add (see figure 16).

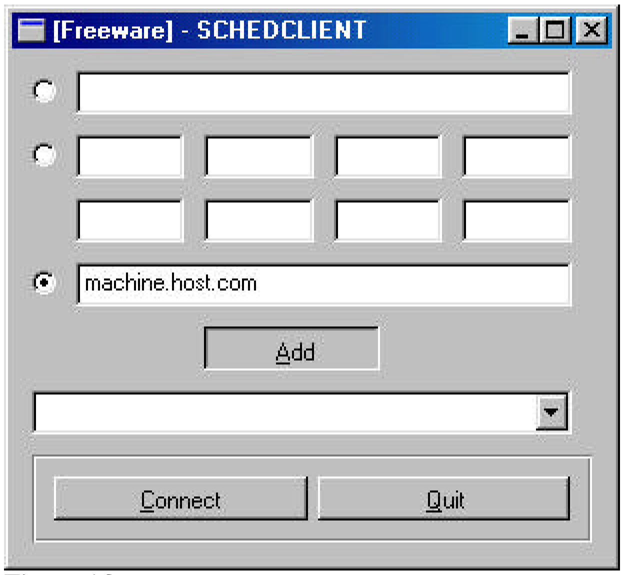


Figure 16

This process is then repeated adding multiple machines to connect to, these host names are added to the list and are viewable in the list box below the add button (see figure 17).

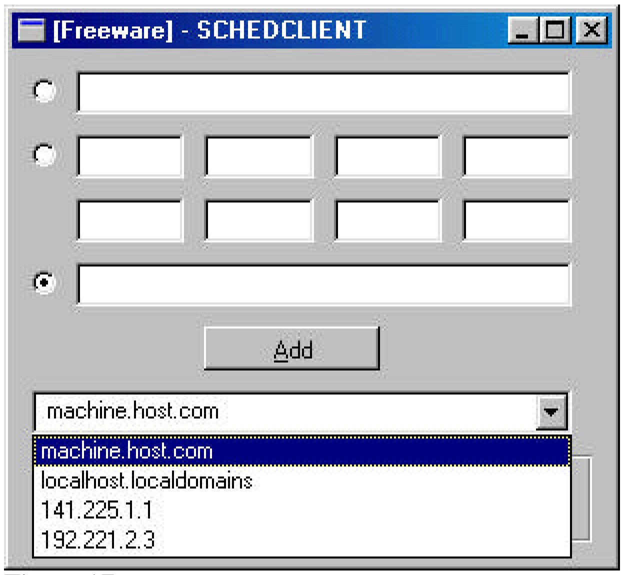


Figure 17

Upon clicking the connect button the user will be presented with a popup similar to the IP range (see figure 18), and clicking on one of the hostnames will open the configuration dialog (see figure 12).

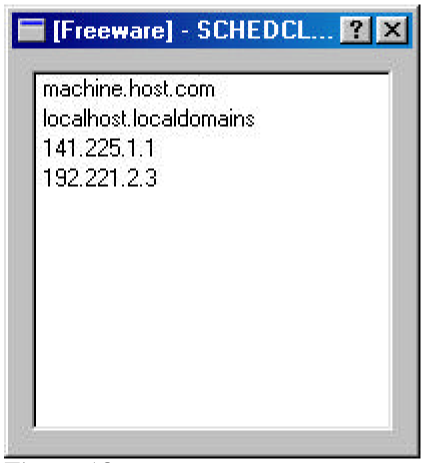


Figure 18

Analysis

The DSS and DSSA were tested on Window 98, 2000, Red Hat Linux, and Debian Linux and in all instances they were compatible in their communications and networking. One of the concerns when running a scheduling program are the memory and CPU usage issues, we ran the DSS on Pentium III at 1GHz with Windows 2000, this machine has 512MB of Ram. Running the DSS on this machine with all the program slots filled with time run programs only yielded a 1,620K-memory usage when on the screen and only 504K used when minimized. CPU usage is 0 when idle and the normal usage for launching a program when the timer times out. On a Linux machine, the usage numbers are similar.

Conclusion

With security on personal and business computers becoming a crucial issue that can no longer be ignored. People and companies have to use more tools to secure their computer, and these tools are becoming more and more complex. A tool like the DSS allow them to use set up their defense once and then rest assured that it will run when needed. For a large company with many computers and many operating systems, this tool will allow them to run the same program on them all regardless of the OS, and also login and reconfigure them from any machine regardless of the OS. For more information, source code, and binaries please see http://issrl.cs.memphis.edu/software.

This work was funded by the Defense Advanced Research Projects Agency (no. F30602-00-2-0514)

References

Booch, G., Rumbaugh, J., and Jacobson, I., *The Unified Modeling Language User Guide,* Addison Wesley, 1999.

Pressman, R., *Software Engineering A Practitioner's Approach*, McGraw Hill, 2001.

TrollTech, “TrollTech QT”, Date Accessed: November 30, 2001, http://www.trolltech.com

Ko, C., Ruschitzka, M., and Levitt, K.,. Execution Monitoring of Security-Critical Programs in Distributed Systems: A Specification-Based Approach. In Proceedings of the 1997 IEEE Symposium on Security and Privacy, Oakland, California, 1997.

Dasgupta, D. and Brian, H. "Mobile security agents for network traffic analysis", Published by the IEEE Computer Society Press in the proceedings of DARPA Information Survivability Conference and Exposition II (DISCEX-II), June 12-14, 2001, Anaheim, California.

Dasgupta, D and Gonzalez, F. "An intelligence decision support system for intrusion detection and response", Information Assurance in Computer Networks, Springer, 2001.

Premkumar T. Devanbu , Philip W-L Fong , Stuart G. Stubblebine. “Techniques for trusted software engineering”, Proceedings of the 1998 international conference on Software engineering April 1998

Moriconi, M., Qian, X., Riemenschneider, R. A., and Gong, L. Secure software architectures. In Proceedings of the 1997 IEEE Symposium on Security and Privacy, pages 84--93, May 1997

Wulf, W. A., Wang, C., and Kienzle, D. A new model of security for distributed systems. Technical Report CS-95-34, University of Virginia, August 1995.